lyrics_comparison

April 30, 2020

1 This is a notebook where I play around with the Python machine learning library `scikit-learn` and `pandas` library to try and differentiate between the albums of two of my favorite artists: Drake and Kanye West.

2 Created by Nurzhan Kanatzhanov.

2.0.1 Standard word processing functions to tokenize and process text

[7]:
```
import math, re
import glob, os
import pandas as pd
from collections import Counter

def tokenize(s):
    """
    Input:
        string s
    Output:
        list of strings
    """
    return s.split()

def preprocess(s, lowercase=True, strip_punctuation=True):
    """
    Input:
        string s
        boolean lowercase
        boolean strip_punctuation
    Return:
        list of strings
    """
    punctuation = '.,?<>:;"\'!%'
    if isinstance(s, str):
        s = tokenize(s)
    if lowercase:
        s = [t.lower() for t in s]
```

```
    if strip_punctuation:
        s = [t.strip(punctuation) for t in s]

    return s

def token_frequency(tokens=None, tf={}, relative=False):
    """
    Input:
        tokens = list of strings or None
        tf = dict or None
        relative = boolean
    Return:
        dictionary of token frequencies
    """
    for t in tokens:
        if t in tf:
            tf[t]+=1
        else:
            tf[t]=1
    if relative:
        total = sum([c for t, c in tf.items()])
        tf = {t:tf[t]/total for t in tf}
    return tf
```

2.0.2 using `glob` module to retrieve files/pathnames of all .txt files (credit to AZLyrics for the lyrics of the artists)

[8]:
```
path = '/Users/nurzhan.kanatzhanov/Desktop/SP2020/Web Portfolio/portfolio/txt/*.
 ↪txt'
filenames = glob.glob(path)
```

2.0.3 setting the variable `TOP_N` to 20 to learn a model on the 20 most frequent words in each artists' album and using them as features (columns) in a pandas DataFrame

[9]:
```
TOP_N = 20

tf = {}
for fn in filenames:
    s = open(fn, 'r').read()
    tf = token_frequency(preprocess(s), tf=tf)

top_f = sorted(tf.items(), key=lambda x:x[1], reverse=True)[:TOP_N]

features = [t[0] for t in top_f]
```

2.0.4 using the os library to split the filenames and give them proper titles to label each album nicely

2.0.5 next, I calculate the relative frequencies of each word (token) in the album and save them in a dictionary, creating vectors for the pandas DataFrame

[10]:
```
labels = [os.path.split(fn)[1][:-4].replace('_', ' ').title() for fn in␣
↪filenames]

vectors = [token_frequency(preprocess(open(f, 'r').read()), tf={},␣
↪relative=True) for f in filenames]

vectors = [{key:v[key] for key in v if key in features} for v in vectors]

vectors_df = pd.DataFrame(vectors, index=labels, columns=features).fillna(0)
```

2.0.6 this is what a 20-feature, 17 album DataFrame looks like, with relative frequencies of each word in each album

[11]:
```
# remove truncation and adjust column width
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)
pd.set_option('display.max_colwidth', -1)
vectors_df
```

[11]:
```
                                             i       the       you  \
Kanye My Beatiful Dark Twisted Fantasy    0.036624  0.043570  0.029362
Drake Take Care                           0.036989  0.032581  0.045286
Kanye College Dropout                     0.039347  0.031830  0.022013
Drake Scorpion                            0.037743  0.030554  0.036844
Drake So Far Gone                         0.048596  0.031977  0.030258
Kanye Jesus Is King                       0.024240  0.061113  0.020826
Kanye Late Registration                   0.033267  0.037939  0.022029
Drake If Youre Reading This Its Too Late  0.042807  0.033295  0.029118
Kanye Graduation                          0.042311  0.030783  0.030910
Kanye Ye                                  0.050205  0.035680  0.030628
Drake More Life                           0.039446  0.029531  0.031023
Drake Nothing Was The Same                0.044998  0.036020  0.029154
Kanye 808S & Heartbreak                   0.045489  0.031164  0.054034
Kanye Yeezus                              0.030827  0.038533  0.025689
Drake Views                               0.046472  0.023131  0.046895
Drake Thank Me Later                      0.044433  0.028781  0.034840
Kanye The Life Of Pablo                   0.046310  0.035860  0.025190

                                            to       and         a  \
Kanye My Beatiful Dark Twisted Fantasy    0.015049  0.019364  0.021995
```

Drake Take Care	0.017285	0.022643	0.015038
Kanye College Dropout	0.022703	0.025234	0.017794
Drake Scorpion	0.020040	0.017344	0.015906
Drake So Far Gone	0.019370	0.028539	0.019943
Kanye Jesus Is King	0.014681	0.010584	0.013315
Kanye Late Registration	0.020583	0.019137	0.016355
Drake If Youre Reading This Its Too Late	0.018561	0.012645	0.021346
Kanye Graduation	0.018115	0.021155	0.013935
Kanye Ye	0.017998	0.013262	0.011683
Drake More Life	0.026759	0.014072	0.018124
Drake Nothing Was The Same	0.020703	0.015422	0.014471
Kanye 808S & Heartbreak	0.019352	0.014828	0.011812
Kanye Yeezus	0.017749	0.016581	0.021485
Drake Views	0.024609	0.018166	0.022708
Drake Thank Me Later	0.016789	0.018935	0.016662
Kanye The Life Of Pablo	0.019030	0.012430	0.016280

	it	me	my \
Kanye My Beatiful Dark Twisted Fantasy	0.011050	0.011682	0.012839
Drake Take Care	0.018927	0.015556	0.008815
Kanye College Dropout	0.010661	0.012272	0.015647
Drake Scorpion	0.013839	0.023275	0.013659
Drake So Far Gone	0.016504	0.010430	0.014900
Kanye Jesus Is King	0.006487	0.010584	0.018436
Kanye Late Registration	0.010236	0.012350	0.011682
Drake If Youre Reading This Its Too Late	0.016821	0.014733	0.012529
Kanye Graduation	0.012921	0.017102	0.013301
Kanye Ye	0.015788	0.010104	0.011999
Drake More Life	0.011834	0.018124	0.015032
Drake Nothing Was The Same	0.025140	0.013415	0.012253
Kanye 808S & Heartbreak	0.017090	0.010304	0.014074
Kanye Yeezus	0.014012	0.012844	0.015180
Drake Views	0.019434	0.021969	0.016265
Drake Thank Me Later	0.018935	0.017420	0.009089
Kanye The Life Of Pablo	0.010450	0.010890	0.014080

	i'm	that	in \
Kanye My Beatiful Dark Twisted Fantasy	0.011261	0.008840	0.014944
Drake Take Care	0.012791	0.017112	0.011235
Kanye College Dropout	0.008360	0.011888	0.009511
Drake Scorpion	0.010873	0.015007	0.008717
Drake So Far Gone	0.021433	0.011117	0.012722
Kanye Jesus Is King	0.007170	0.005463	0.009560
Kanye Late Registration	0.009123	0.012239	0.009791
Drake If Youre Reading This Its Too Late	0.022390	0.011485	0.011021
Kanye Graduation	0.011274	0.008994	0.010008
Kanye Ye	0.005684	0.013262	0.007262

Drake More Life	0.014712	0.010021	0.007569
Drake Nothing Was The Same	0.009295	0.014049	0.008873
Kanye 808S & Heartbreak	0.013069	0.014577	0.011561
Kanye Yeezus	0.014012	0.009809	0.021018
Drake Views	0.011935	0.011301	0.009717
Drake Thank Me Later	0.011613	0.015021	0.009215
Kanye The Life Of Pablo	0.010670	0.009020	0.015070

	on	like	know \
Kanye My Beatiful Dark Twisted Fantasy	0.005578	0.006104	0.009051
Drake Take Care	0.008297	0.007864	0.012358
Kanye College Dropout	0.008283	0.009587	0.006213
Drake Scorpion	0.010784	0.008088	0.007728
Drake So Far Gone	0.016619	0.007106	0.006418
Kanye Jesus Is King	0.019802	0.008194	0.003073
Kanye Late Registration	0.008011	0.007343	0.004673
Drake If Youre Reading This Its Too Late	0.011717	0.009049	0.010093
Kanye Graduation	0.008487	0.008741	0.007347
Kanye Ye	0.014209	0.006315	0.010420
Drake More Life	0.010448	0.010981	0.010341
Drake Nothing Was The Same	0.013309	0.010140	0.008662
Kanye 808S & Heartbreak	0.006534	0.007288	0.015330
Kanye Yeezus	0.012377	0.006539	0.007940
Drake Views	0.011829	0.013942	0.008344
Drake Thank Me Later	0.007700	0.006564	0.007069
Kanye The Life Of Pablo	0.007370	0.006600	0.007700

	for	we	up \
Kanye My Beatiful Dark Twisted Fantasy	0.008630	0.007262	0.006104
Drake Take Care	0.007692	0.007087	0.006568
Kanye College Dropout	0.006443	0.006596	0.009434
Drake Scorpion	0.014558	0.004134	0.006380
Drake So Far Gone	0.005616	0.004928	0.006074
Kanye Jesus Is King	0.006487	0.022875	0.007170
Kanye Late Registration	0.006676	0.010570	0.006453
Drake If Youre Reading This Its Too Late	0.007193	0.007541	0.009745
Kanye Graduation	0.004180	0.007601	0.006714
Kanye Ye	0.004105	0.006947	0.008525
Drake More Life	0.007143	0.008742	0.010554
Drake Nothing Was The Same	0.008556	0.008767	0.005915
Kanye 808S & Heartbreak	0.003770	0.005529	0.003267
Kanye Yeezus	0.003270	0.011443	0.008174
Drake Views	0.009400	0.005703	0.005598
Drake Thank Me Later	0.007574	0.005049	0.005933
Kanye The Life Of Pablo	0.007920	0.009130	0.008030

	they	your

```
Kanye My Beatiful Dark Twisted Fantasy    0.005367  0.006630
Drake Take Care                           0.010198  0.008297
Kanye College Dropout                     0.004679  0.008360
Drake Scorpion                            0.009795  0.008447
Drake So Far Gone                         0.005616  0.006189
Kanye Jesus Is King                       0.005463  0.008877
Kanye Late Registration                   0.006453  0.005785
Drake If Youre Reading This Its Too Late  0.008701  0.004408
Kanye Graduation                          0.005067  0.004687
Kanye Ye                                  0.008841  0.004736
Drake More Life                           0.006716  0.006610
Drake Nothing Was The Same                0.004859  0.004753
Kanye 808S & Heartbreak                   0.003770  0.009299
Kanye Yeezus                              0.006305  0.011910
Drake Views                               0.011301  0.006232
Drake Thank Me Later                      0.007448  0.006943
Kanye The Life Of Pablo                   0.005280  0.007590
```

2.0.7 using scikit-learn's KMeans to learn 2 clusters from the data

[12]:
```
from sklearn.cluster import KMeans

n_clusters = 2
kmeans = KMeans(n_clusters = n_clusters, random_state = 0).fit(vectors_df)
```

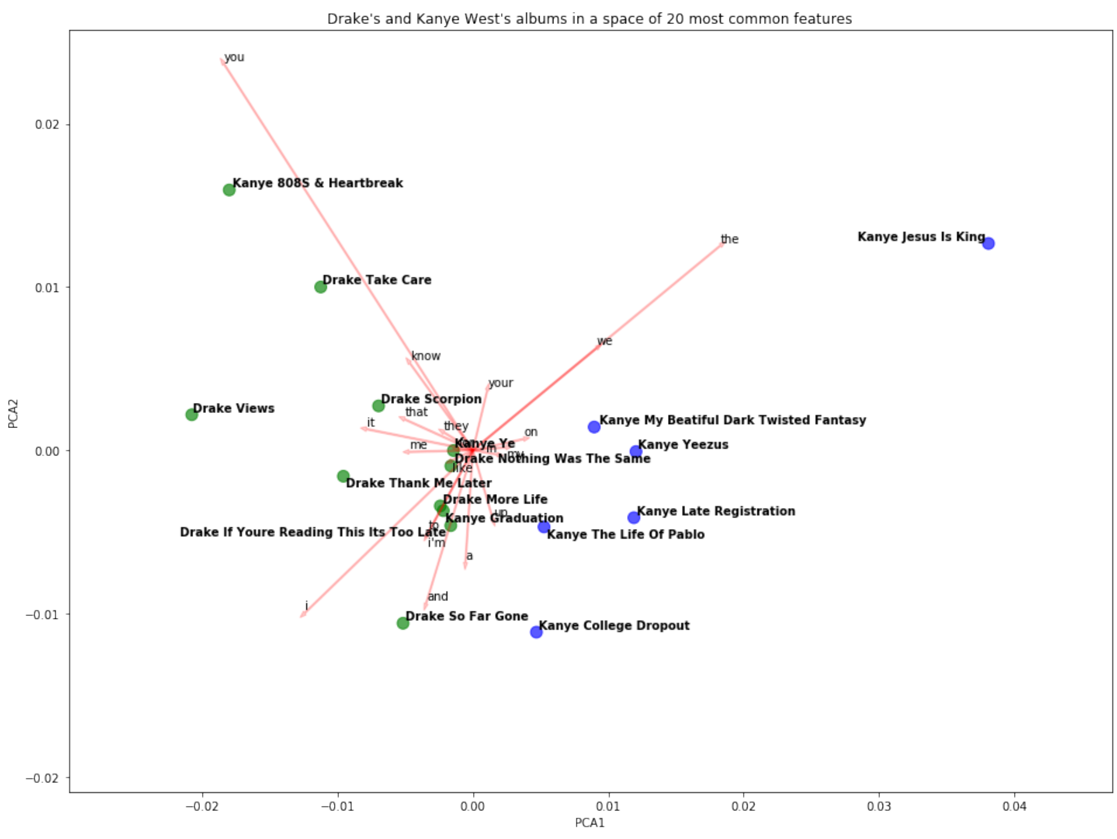
2.0.8 projecting all 20 features onto a 2-dimensional space using scikit-learn's principal component analysis (PCA).

[13]:
```
from sklearn.decomposition import PCA

pca = PCA(n_components=2)
transformed = pca.fit_transform(vectors_df)

x = transformed[:,0]
y = transformed[:,1]
```

2.0.9 finally, using the matplotlib Python plotting library and an automatic label placement library adjustText (link) to make a scatter plot of the 17 albums on a 2D transformed space.

[18]:
```
import matplotlib.pyplot as plt
from adjustText import adjust_text

col_dict = {0:'green', 1:'blue'}
cols = [col_dict[l] for l in kmeans.labels_]
plt.figure(figsize=(16,12))
```

```
plt.scatter(x,y, c=cols, s=100, alpha=.65)
texts = []
for i, l in enumerate(labels):
    texts.append(plt.text(x[i],y[i], l, weight='bold'))
arrows = []
for i, c in enumerate(pca.components_.transpose()):
    plt.arrow(0,0, c[0]/30, c[1]/30, alpha=.2, width=.0001, color="red")
    arrows.append(plt.text(c[0]/30, c[1]/30, features[i]))
plt.xlabel('PCA1')
plt.ylabel('PCA2')
plt.title("Drake's and Kanye West's albums in a space of {} most common␣
↪features".format(TOP_N))
adjust_text(texts)
adjust_text(arrows)
plt.show()
```



2.0.10 as we see from the plot above, 6 out of 9 Kanye West albums got clustered together in blue (*Jesus is king*, *My Beautiful Dark Twisted Fantasy*, *Yeezus*, *Late Registration*, *The Life of Pablo*, and *College Dropout*). However, 3 of them (*808s & Hearbreak*, *Ye*, and *Graduation*) were clustered with Drake's albums. It is good to note that all of Drake's albums were clustered together (in green), showing common similarities in the rapper's word choice.

2.0.11 Now, let's try something different. As expected in any text corpus, the most common features would be short stopwords like *you*, *the*, *it*, *I*, etc. Let's see what would happen if we try to exclude these words and play around with `gensim`, an unsupervised topic modeling and language processing library.

implementing a helper function that would filter out the stopwords

[15]:
```
def get_texts(filenames, stop_words):
    for fn in filenames:
        text = open(fn, 'r').read()
        text = [t for t in preprocess(text) if t not in stop_words]
        yield(text)
```

[74]:
```
NUM_TOPICS = 5
TOPN = 15
STOP = 100
```

2.0.12 One of the strategies to get the stopwords is too find the 100 most frequent words

[75]:
```
freqs = {}
for file in filenames:
    freqs = token_frequency(preprocess(open(file, 'r').read()), tf=freqs)
stop_words = sorted(freqs, key=freqs.__getitem__, reverse=True)[:STOP]
```

2.0.13 Make a `gensim` `Dictionary` first to map between words and their integer ID's, and then convert each document into the bag-of-words (BoW) format.

2.0.14 Next, I use the `gensim` Latent Dirichlet Allocation (LDA) algorithm for topic modeling

[76]:
```
from gensim import corpora, models, similarities

dictionary = corpora.Dictionary(get_texts(filenames, stop_words))
corpus = [dictionary.doc2bow(text) for text in get_texts(filenames, stop_words)]
lda = models.LdaModel(corpus, id2word=dictionary, num_topics=NUM_TOPICS)

corpus_lda = lda[corpus]
```

2.0.15 For visualization, this is what the top 15 words in each topic would be:

[77]:
```
for topic in range(NUM_TOPICS):
    tt = lda.get_topic_terms(topic, topn=TOPN)
    top_words = [dictionary[t] for t, w in tt]
    top_words = ', '.join(top_words)
    print('Topic {:>2d}: {}'.format(topic, top_words))
```

```
Topic  0: who, look, them, new, through, even, will, everything, real, em,
think, come, did, always, god
Topic  1: why, look, things, money, think, where, or, thing, real, us, only,
than, god, even, night
Topic  2: us, as, who, money, said, over, much, them, only, think, god, new,
i'll, things, even
Topic  3: even, them, where, everything, over, us, think, mind, real, only, or,
as, em, come, i'll
Topic  4: only, them, over, where, as, new, or, think, everything, us, money,
always, i'll, said, night
```

2.0.16 This is each albums distribution of topics by percentage:

[78]:
```
for i, label in enumerate(labels):
    topics = sorted(corpus_lda[i], key = lambda x:x[1], reverse=True)
    topics = ['Topic {} ({:2.2f}%)'.format(t[0], t[1]*100) for t in topics]
    topics = ', '.join(topics)
    print('{}:\n{}\n'.format(label, topics))
```

```
Kanye My Beatiful Dark Twisted Fantasy:
Topic 2 (68.28%), Topic 1 (28.41%), Topic 0 (3.30%)

Drake Take Care:
Topic 3 (87.61%), Topic 0 (6.78%), Topic 2 (3.89%), Topic 1 (1.70%)

Kanye College Dropout:
Topic 2 (70.80%), Topic 1 (24.85%), Topic 3 (2.46%), Topic 4 (1.89%)

Drake Scorpion:
Topic 3 (64.06%), Topic 0 (15.62%), Topic 1 (11.71%), Topic 2 (8.23%)

Drake So Far Gone:
Topic 1 (89.16%), Topic 3 (7.61%), Topic 4 (2.16%), Topic 0 (1.06%)

Kanye Jesus Is King:
Topic 2 (99.92%)

Kanye Late Registration:
Topic 3 (93.90%), Topic 0 (3.46%), Topic 1 (2.61%)
```

```
Drake If Youre Reading This Its Too Late:
Topic 1 (88.01%), Topic 3 (7.78%), Topic 0 (3.75%)

Kanye Graduation:
Topic 3 (62.37%), Topic 0 (20.73%), Topic 1 (16.89%)

Kanye Ye:
Topic 1 (99.11%)

Drake More Life:
Topic 0 (67.26%), Topic 1 (30.99%)

Drake Nothing Was The Same:
Topic 2 (77.77%), Topic 0 (20.26%), Topic 1 (1.95%)

Kanye 808S & Heartbreak:
Topic 0 (92.71%), Topic 1 (7.25%)

Kanye Yeezus:
Topic 3 (74.88%), Topic 1 (18.16%), Topic 0 (6.94%)

Drake Views:
Topic 1 (53.94%), Topic 3 (36.03%), Topic 2 (10.02%)

Drake Thank Me Later:
Topic 0 (64.11%), Topic 3 (35.36%)

Kanye The Life Of Pablo:
Topic 2 (57.53%), Topic 1 (23.70%), Topic 0 (11.88%), Topic 3 (6.89%)
```

2.0.17 I am also going to use gensim's similarities class that "computes similarities across a collection of documents in the Vector Space Model." This will connect the most similar albums between Drake and Kanye West.

[79]:

```python
similarity_index = similarities.SparseMatrixSimilarity(corpus_lda,␣
↪num_features=NUM_TOPICS)

print('Most similar texts:\n')
for i, label in enumerate(labels):
    sim = similarity_index[corpus_lda[i]]
    sim_labels = sorted(zip(sim, labels), reverse=True)
    sim_print = [l for s, l in sim_labels][1:4]
    sim_print = ', '.join(sim_print)
    print('{}: {}\n'.format(label, sim_print))
```

```
Most similar texts:
```

Kanye My Beatiful Dark Twisted Fantasy: Kanye College Dropout, Kanye The Life Of Pablo, Kanye Jesus Is King

Drake Take Care: Kanye Late Registration, Drake Scorpion, Kanye Yeezus

Kanye College Dropout: Kanye College Dropout, Kanye The Life Of Pablo, Kanye Jesus Is King

Drake Scorpion: Kanye Yeezus, Drake Take Care, Kanye Graduation

Drake So Far Gone: Drake If Youre Reading This Its Too Late, Kanye Ye, Drake Views

Kanye Jesus Is King: Drake Nothing Was The Same, Kanye My Beatiful Dark Twisted Fantasy, Kanye College Dropout

Kanye Late Registration: Drake Take Care, Kanye Yeezus, Drake Scorpion

Drake If Youre Reading This Its Too Late: Drake So Far Gone, Kanye Ye, Drake Views

Kanye Graduation: Drake Scorpion, Kanye Yeezus, Kanye Late Registration

Kanye Ye: Drake So Far Gone, Drake If Youre Reading This Its Too Late, Drake Views

Drake More Life: Kanye 808S & Heartbreak, Drake Thank Me Later, Drake If Youre Reading This Its Too Late

Drake Nothing Was The Same: Kanye Jesus Is King, Kanye My Beatiful Dark Twisted Fantasy, Kanye The Life Of Pablo

Kanye 808S & Heartbreak: Drake More Life, Drake Thank Me Later, Kanye Graduation

Kanye Yeezus: Drake Scorpion, Kanye Late Registration, Drake Take Care

Drake Views: Drake If Youre Reading This Its Too Late, Drake So Far Gone, Kanye Ye

Drake Thank Me Later: Kanye 808S & Heartbreak, Drake More Life, Kanye Graduation

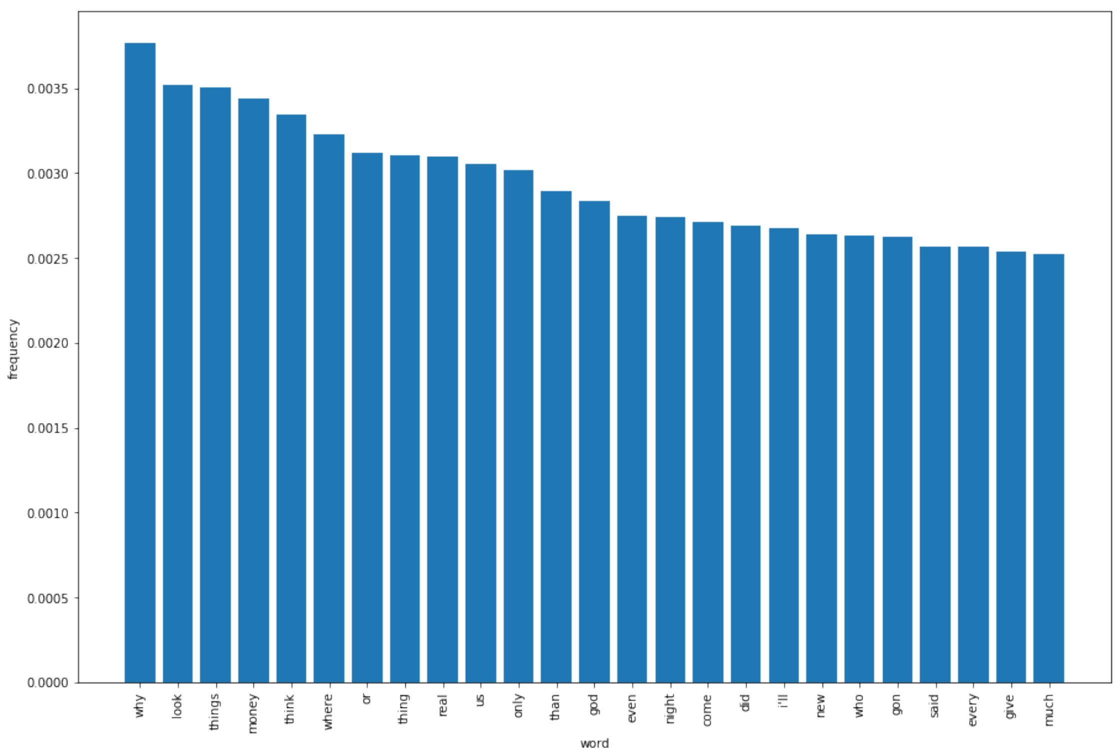
Kanye The Life Of Pablo: Kanye My Beatiful Dark Twisted Fantasy, Kanye College Dropout, Drake Nothing Was The Same

2.0.18 Let's visualize 25 most common words:

[80]:
```
import matplotlib.pyplot as plt

plt.figure(figsize=(15,10))
print(lda.get_topics()[0])
tokens, y = zip(*lda.get_topic_terms(1, topn=25))
tokens = [dictionary[t] for t in tokens]
x = list(range(25))
plt.bar(x,y, tick_label=tokens)
plt.xticks(rotation='vertical')
plt.xlabel('word')
plt.ylabel('frequency')
```

```
[2.0291578e-04 2.8627848e-05 2.8341370e-05 … 2.7051228e-05 2.4993044e-05
 3.8262078e-05]
```

[80]:
```
Text(0, 0.5, 'frequency')
```



2.0.19 Now, let's create vectors for the new pandas DataFrame with the distribution of topics of the 17 albums

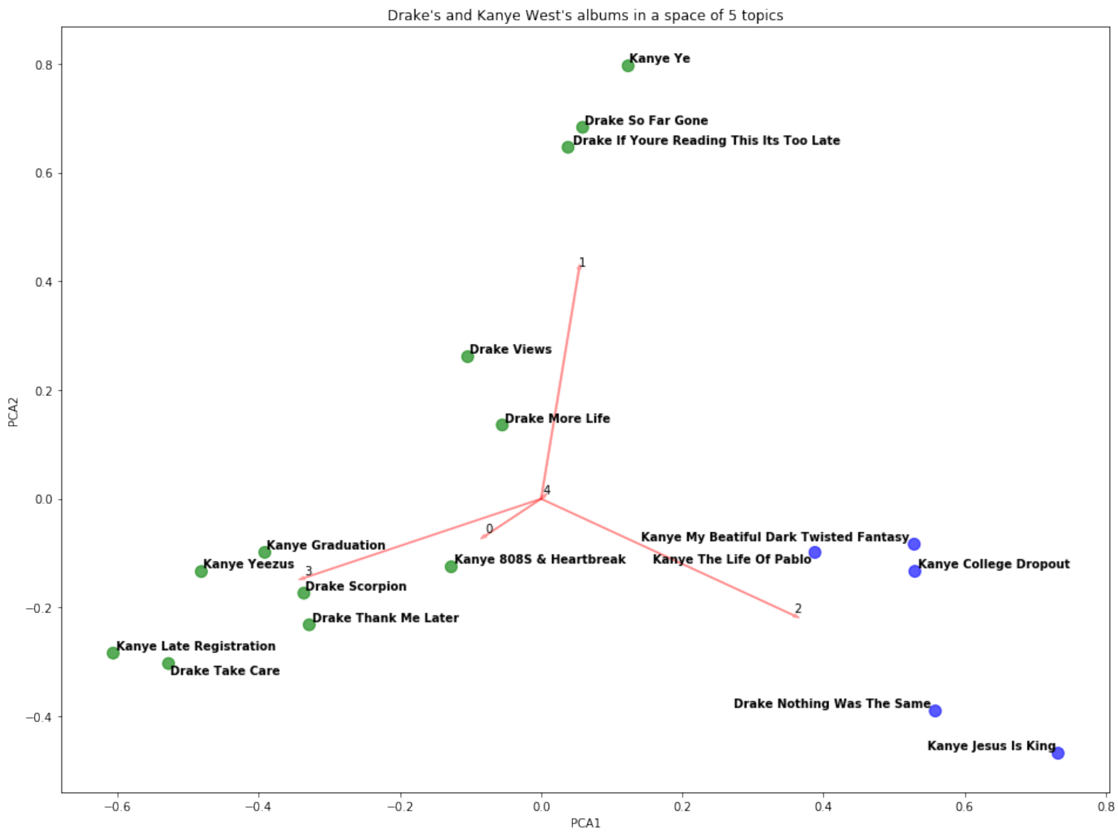
[81]:
```
topics = list(range(NUM_TOPICS))
vectors = [{index:ratio for index, ratio in v} for v in corpus_lda]

vectors_df = pd.DataFrame(vectors, index=labels, columns=topics).fillna(0)
```

[82]:
```
n_clusters=2
kmeans = KMeans(n_clusters=n_clusters, random_state=0).fit(vectors_df)
kmeans.labels_
```

[82]:
```
array([1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1], dtype=int32)
```

[83]:
```
pca = PCA(n_components=2)
transformed = pca.fit_transform(vectors_df)

x = transformed[:,0]
y = transformed[:,1]
```

[84]:
```
col_dict = {0:'green', 1:'blue'}
cols = [col_dict[l] for l in kmeans.labels_]
plt.figure(figsize=(16,12))
plt.scatter(x,y, c=cols, s=100, alpha=.65)
texts = []
for i, l in enumerate(labels):
    texts.append(plt.text(x[i],y[i], l, weight='bold'))
arrows = []
for i, c in enumerate(pca.components_.transpose()):
    plt.arrow(0,0, c[0]/2, c[1]/2, alpha=.3, width=.002, color="red")
    arrows.append(plt.text(c[0]/2, c[1]/2, topics[i]))
plt.xlabel('PCA1')
plt.ylabel('PCA2')
plt.title("Drake's and Kanye West's albums in a space of {} topics".
 ↪format(NUM_TOPICS))
adjust_text(texts)
adjust_text(arrows)
plt.show()
```



2.0.20 Now, let's try something interesting: bring in another artist from a completely different genre of music. After some thought, I decided to include AC/DC, a hard rock band, to see how well the classification will work between different genres/styles of music.

[99]:
```
path_acdc = '/Users/nurzhan.kanatzhanov/Desktop/SP2020/Web Portfolio/portfolio/
 ↪txt/acdc/*.txt'
filenames_acdc = glob.glob(path_acdc)
filenames_acdc.extend(filenames)
```

[104]:
```
NUM_TOPICS = 10

freqs = {}
for file in filenames_acdc:
    freqs = token_frequency(preprocess(open(file, 'r').read()), tf=freqs)
stop_words = sorted(freqs, key=freqs.__getitem__, reverse=True)[:STOP]

from gensim import corpora, models, similarities

dictionary = corpora.Dictionary(get_texts(filenames_acdc, stop_words))
```

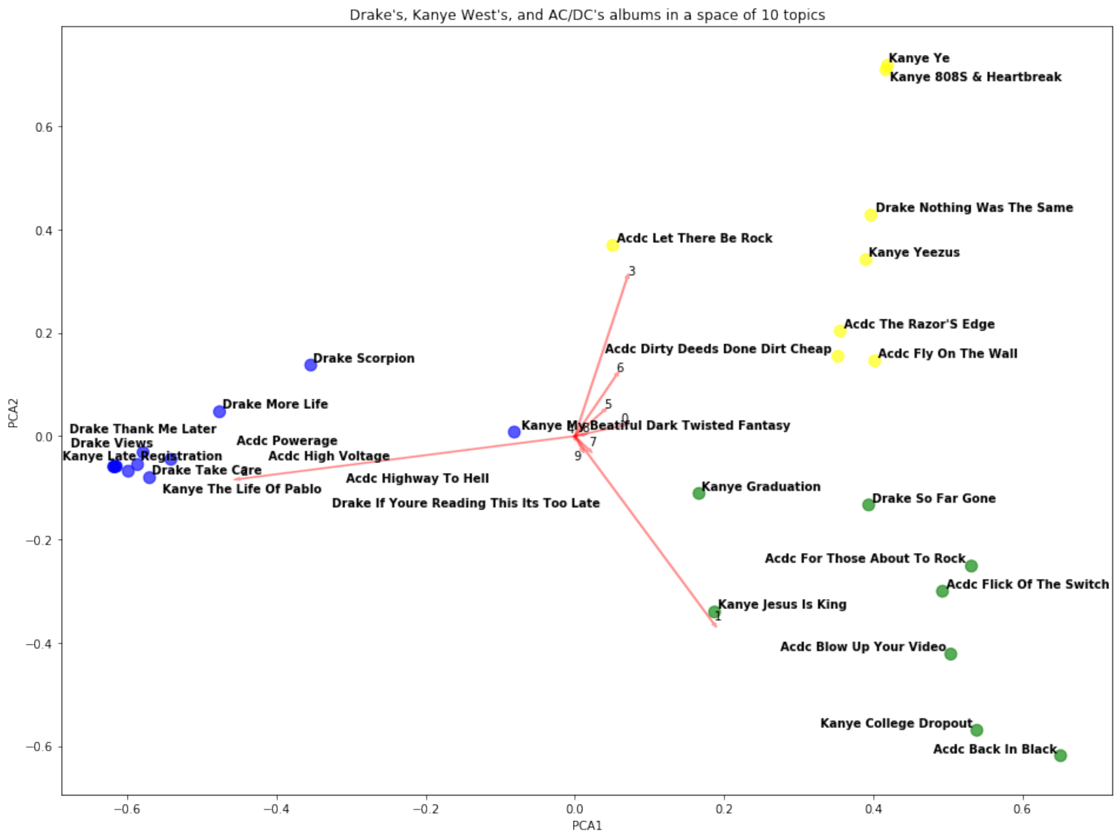
```
corpus = [dictionary.doc2bow(text) for text in get_texts(filenames_acdc,␣
 ↪stop_words)]
lda = models.LdaModel(corpus, id2word=dictionary, num_topics=NUM_TOPICS)

corpus_lda = lda[corpus]
```

[105]:
```
topics = list(range(NUM_TOPICS))
vectors = [{index:ratio for index, ratio in v} for v in corpus_lda]
labels = [os.path.split(fn)[1][:-4].replace('_', ' ').title() for fn in␣
 ↪filenames_acdc]

vectors_df = pd.DataFrame(vectors, index=labels, columns=topics).fillna(0)
```

2.0.21 Trying to cluster into 3 groups now

[106]:
```
n_clusters=3
kmeans = KMeans(n_clusters=n_clusters, random_state=0).fit(vectors_df)

pca = PCA(n_components=2)
transformed = pca.fit_transform(vectors_df)

x = transformed[:,0]
y = transformed[:,1]
```

[107]:
```
col_dict = {0:'green', 1:'blue', 2:'yellow'}
cols = [col_dict[l] for l in kmeans.labels_]
plt.figure(figsize=(16,12))
plt.scatter(x,y, c=cols, s=100, alpha=.65)
texts = []
for i, l in enumerate(labels):
    texts.append(plt.text(x[i],y[i], l, weight='bold'))
arrows = []
for i, c in enumerate(pca.components_.transpose()):
    plt.arrow(0,0, c[0]/2, c[1]/2, alpha=.3, width=.002, color="red")
    arrows.append(plt.text(c[0]/2, c[1]/2, topics[i]))
plt.xlabel('PCA1')
plt.ylabel('PCA2')
plt.title("Drake's, Kanye West's, and AC/DC's albums in a space of {} topics".
 ↪format(NUM_TOPICS))
adjust_text(texts)
adjust_text(arrows)
plt.show()
```



2.0.22 Seems like the addition of AC/DC really does not show any key differences in topic modeling between AC/DC, Kanye West, and Drake!

```
[ ]:
```